

SimpleShell.java

```

1 import java.io.*;
5
6
7 class SimpleShell
8 {
9
10     //displays history of commands from an array list
11     private static void historyOutput(ArrayList<String> history)
12     {
13         for(int i = 0; i < history.size(); i++)
14             System.out.println(i + " " + history.get(i));
15     }
16
17     //returns the previous command from the array list
18     private static String returnPrevious(ArrayList<String> history)
19     {
20         return history.get(history.size() - 2);
21     }
22
23     public static void main(String[] args) throws IOException
24     {
25         String commandLine;
26         BufferedReader console = new BufferedReader(new InputStreamReader(System.in));
27         ArrayList<String> historyList = new ArrayList<String>();
28
29         while(true)
30         {
31
32             //read what the user has entered
33             System.out.println("jsh>");
34             commandLine = console.readLine();
35
36             //Creates a history Array List storing all the commands inserted by the user so
37             far
38             historyList.add(commandLine);
39
40             int commNum;
41
42             //CHECKS FOR PART 3 COMMANDS
43             //If user input is "history" it displays all previous user inputs
44             if(commandLine.equals("history"))
45             {
46                 historyOutput(historyList);
47                 //System.out.println("The size is: " + historyList.size());
48             }
49             else
50             {
51                 //command to return the previous command from the user
52                 if(commandLine.equals("!!") && historyList.size() >= 1)
53                 {
54                     commandLine = returnPrevious(historyList);
55                 }
56                 else
57                 {
58                     //If there is no previous command to return to it displays an error
59                     if(commandLine.equals("!!") && historyList.size() < 1)
60                     {
61                         System.out.println("Unable to run previous command for there is no
62 previous command.");
63                     }
64                 }
65             }
66         }
67     }
68 }

```

SimpleShell.java

```
59         continue;
60     }
61     else
62         //command to go to a specific 'i'th command
63         if(commandLine.charAt(0) == '!' && commandLine.length() == 2)
64         {
65             commNum = Integer.parseInt(commandLine.substring(1));
66             commandLine = historyList.get(commNum - 1);
67         }
68
69
70
71     //if the user has entered a return, just loop again
72     if (commandLine.equals(""))
73         continue;
74
75     //It separates the words of the user inputs on every "space" there is
76     String[] parameterArray = commandLine.split(" ");
77     System.out.println("Output String Array: " + Arrays.toString(parameterArray));
78
79     //System.out.println("Value at index 1 of array:" + parameterArray[1]);
80
81     //It adds all the elements of the parameter String Array into a String Array List
82     List<String> parameterList = Arrays.asList(parameterArray);
83
84
85
86
87     //System.out.println("user input: " + commandLine);
88
89     //System.out.println(System.getProperty("User current location is: " +
90 "user.dir"));
91     try
92     {
93
94         //created a ProcessBuilder objected passing the parameterList to the
95         constructor
96         ProcessBuilder pb = new ProcessBuilder(parameterList);
97         System.out.println("The String Array List Command Was: " + pb.command() +
98 "\n");
99
100         //System.out.println(System.getProperty("User current location is: " +
101 "user.dir"));
102
103         //CHECKS FOR PART 2 COMMANDS
104         if (commandLine.equals("cd") || commandLine.equals("cd ~"))
105             pb.directory(new File(System.getProperty("user.home")));
106         else
107             if(commandLine.equals("cd /"))
108                 pb.directory(new File("/")); // Go to root directory case
109             else
110                 //Checks if first argument is "cd" and second argument does not
111                 contains a '/'
112                 if((parameterArray[0].equals("cd")) && (parameterArray[1].indexOf('/')
113 < 0))
```

SimpleShell.java

```
110      {
111          //calls the word line after the "cd", e.g. cd music would set
          directory to music
112          pb.directory(new File(System.getProperty("user.dir") + "/" +
          parameterArray[1]));
113
114          //System.out.println("Parameter with no slash" + ": is " +
          parameterArray[1]);
115      }
116      else
117          //Checks if first argument is "cd" and second argument contains a
          '/'
118          if((parameterArray[0].equals("cd")) &&
          (parameterArray[1].indexOf('/') >= 0))
119          {
120              System.out.println(System.getProperty("user.dir") + "/" +
          parameterArray[1]);
121
122              pb.directory(new File(parameterArray[1]));
123
124          }
125
126
127          //System.out.println("The file Path was: " + filePath);
128
129
130          //starts the process
131          Process process = pb.start();
132
133
134          //obtains output stream
135          InputStream is = process.getInputStream();
136          InputStreamReader isr = new InputStreamReader(is);
137          BufferedReader br = new BufferedReader(isr);
138
139          //outputs the contents returned by the command
140          String line;
141          while ( (line = br.readLine()) != null)
142              System.out.println(line);
143
144          br.close();
145
146      }
147      //If invalid command, it displays the error message and waits for further commands
      from the user.
148      catch(IOException ex)
149      {
150
151          System.out.println("\n" + ex.toString());
152          System.out.println("Possible invalid command. Please try again..." + "\n");
153          continue;
154      }
155
156
157
158
159
```

SimpleShell.java

```
160
161
162     }
163
164
165     }
166
167 }
168
```

Output

Part 1:

```
jsh>
ls
Output String Array: [ls]
The String Array List Command Was: [ls]

CSC320_Assigment1.java
MyFirstProgram.class
MyFirstProgram.java
SimpleShell.class
SubFolder
jsh>
ls
Output String Array: [ls]
The String Array List Command Was: [ls]

CSC320_Assigment1.java
MyFirstProgram.class
MyFirstProgram.java
SimpleShell.class
SubFolder
```

Running the ls command within the terminal from the virtual computer

Part 2:

```
^Cvitorjpc10@ubuntu:~/MyJavaDirectory$ ls
CSC320_Assigment1.java  MyFirstProgram.java  SubFolder
MyFirstProgram.class    SimpleShell.class
vitorjpc10@ubuntu:~/MyJavaDirectory$ cd SubFolder
vitorjpc10@ubuntu:~/MyJavaDirectory/SubFolder$
```

Desired Output

```
jsh>
ls
Output String Array: [ls]
The String Array List Command Was: [ls]

CSC320_Assigment1.java
MyFirstProgram.class
MyFirstProgram.java
SimpleShell.class
SubFolder
jsh>
cd SubFolder
Output String Array: [cd, SubFolder]
The String Array List Command Was: [cd, SubFolder]

java.io.IOException: Cannot run program "cd" (in directory "/home/vitorjpc10/My
JavaDirectory/SubFolder"): error=2, No such file or directory
Possible invalid command. Please try again...

jsh>
```

Actual output

Unfortunately, even after the use of the *.directory* method (starting a line 101) I was unable to set the new directory even after considering the use of *getProperty* and calling all the parameters accordingly to the type of user command input.

Part 3:

```
jsh>
history
0    AAA
1    BBB
2    CCC
3    DDD
4    ls
5    history
Output String Array: [history]
The String Array List Command Was: [history]

java.io.IOException: Cannot run program "history": error=2, No such file or dir
ectory
Possible invalid command. Please try again...

jsh>
```

Calling history command

```
jsh>
ls
Output String Array: [ls]
The String Array List Command Was: [ls]

CSC320_Assignment1.java
MyFirstProgram.class
MyFirstProgram.java
SimpleShell.class
SubFolder
jsh>
!!
Output String Array: [ls]
The String Array List Command Was: [ls]

CSC320_Assignment1.java
MyFirstProgram.class
MyFirstProgram.java
SimpleShell.class
SubFolder
jsh>
```

Calling the !! command

```
jsh>
history
0  AAA
1  BBB
2  CCC
3  DDD
4  ls
5  history
6  !!
7  ls
8  !!
9  history
Output String Array: [history]
The String Array List Command Was: [history]

java.io.IOException: Cannot run program "history": error=2, No such file or directory
Possible invalid command. Please try again...

jsh>
!8
Output String Array: [ls]
The String Array List Command Was: [ls]

CSC320_Assignment1.java
MyFirstProgram.class
MyFirstProgram.java
SimpleShell.class
SubFolder
jsh>
```

Calling the !# command (Where # is any number within the command history). In this case !8 was called.